

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

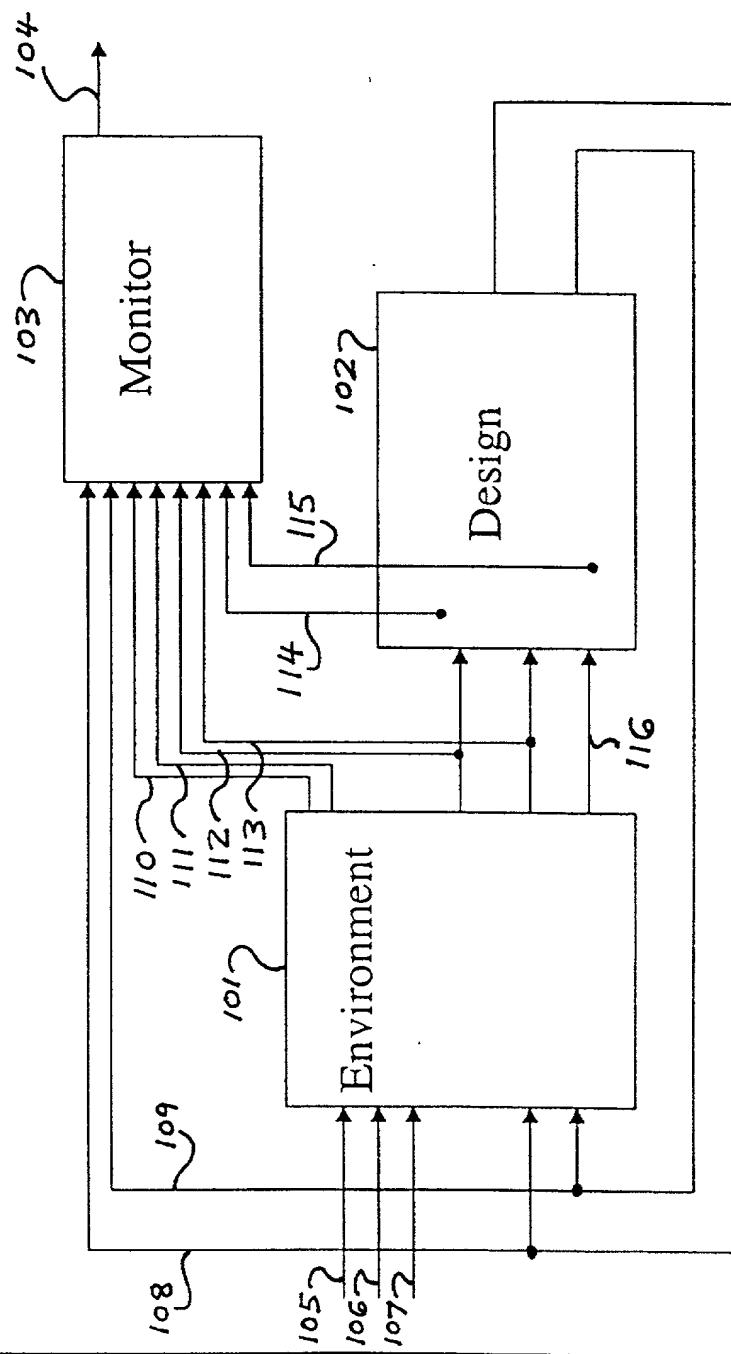
J.H. Kukula, et al.

06816.0036

1/50

FIGURE 1 G.1

# Monitor and Environment



METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

2/50

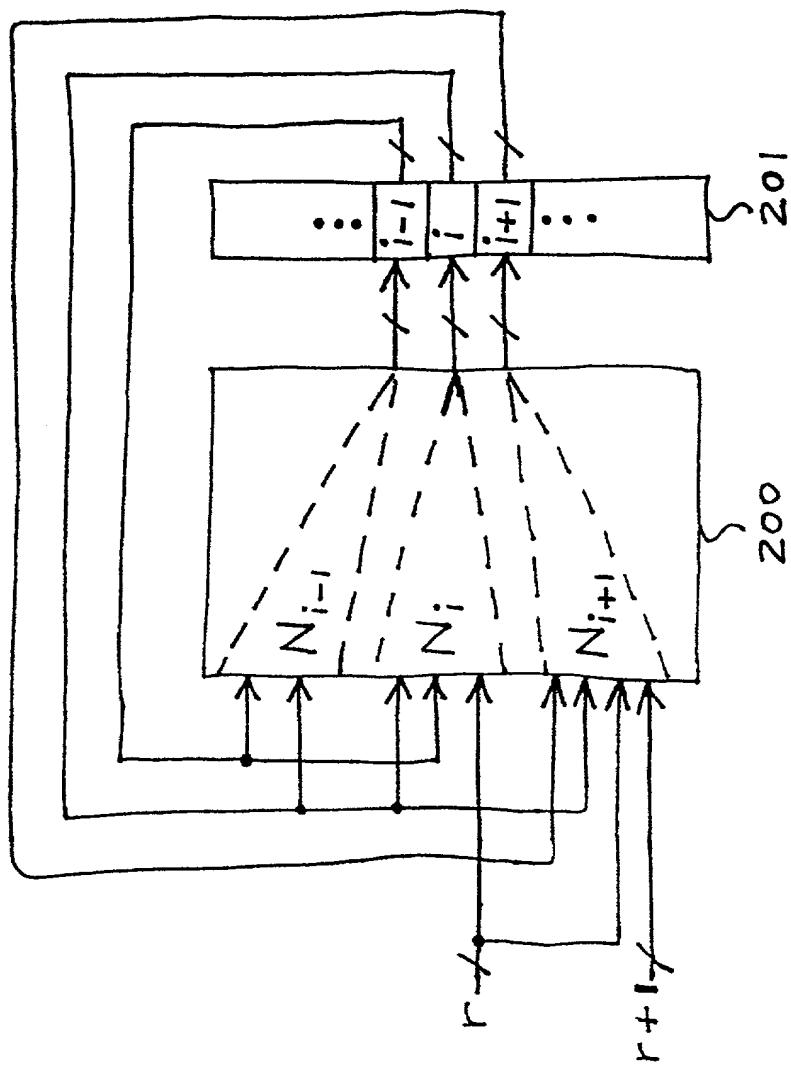


Figure 2

**METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION**

J.H. Kukula, et al.

06816.0036

3/50

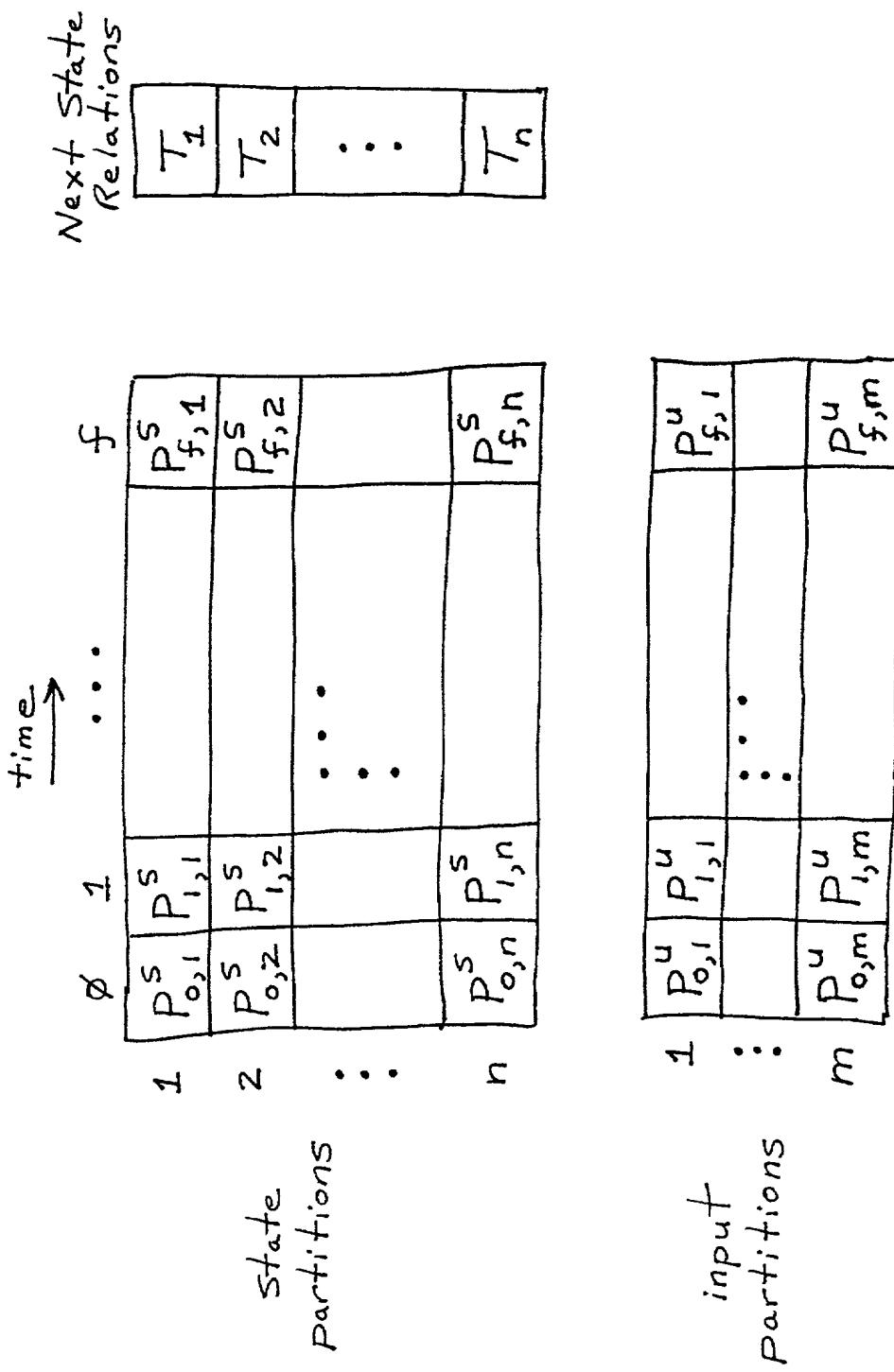


FIGURE 3

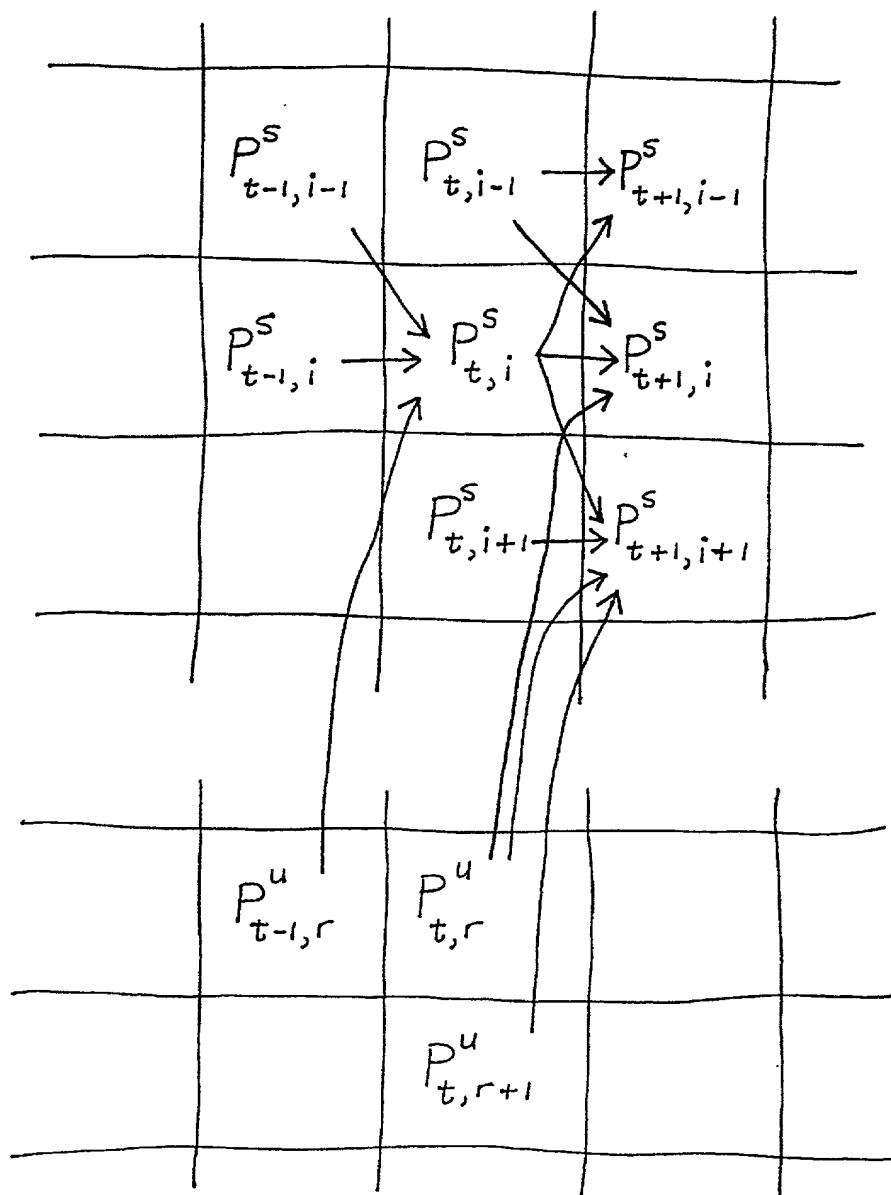
METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

4/50

FIG. 4 A



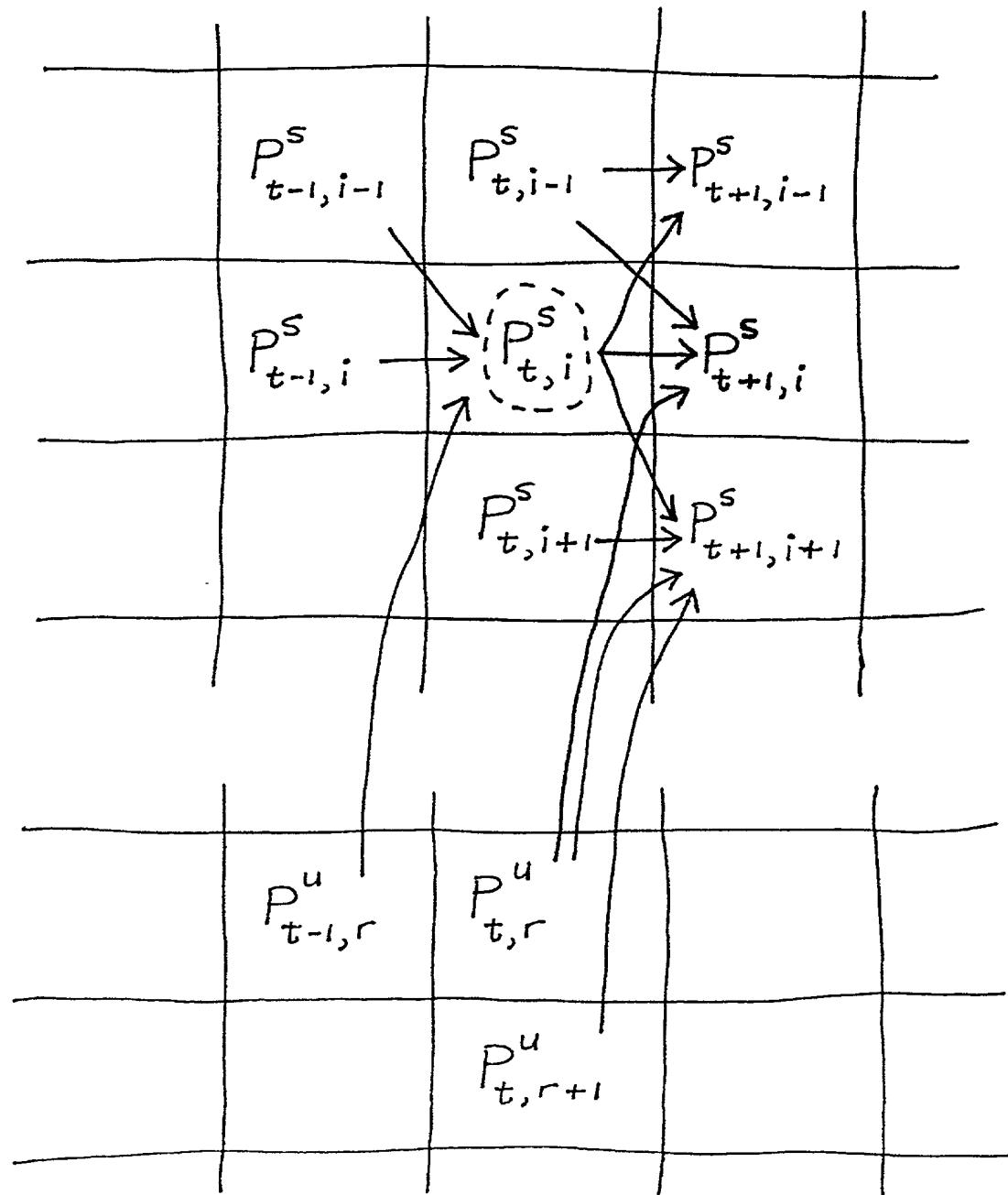
METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

5/50

FIG. 4B



METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

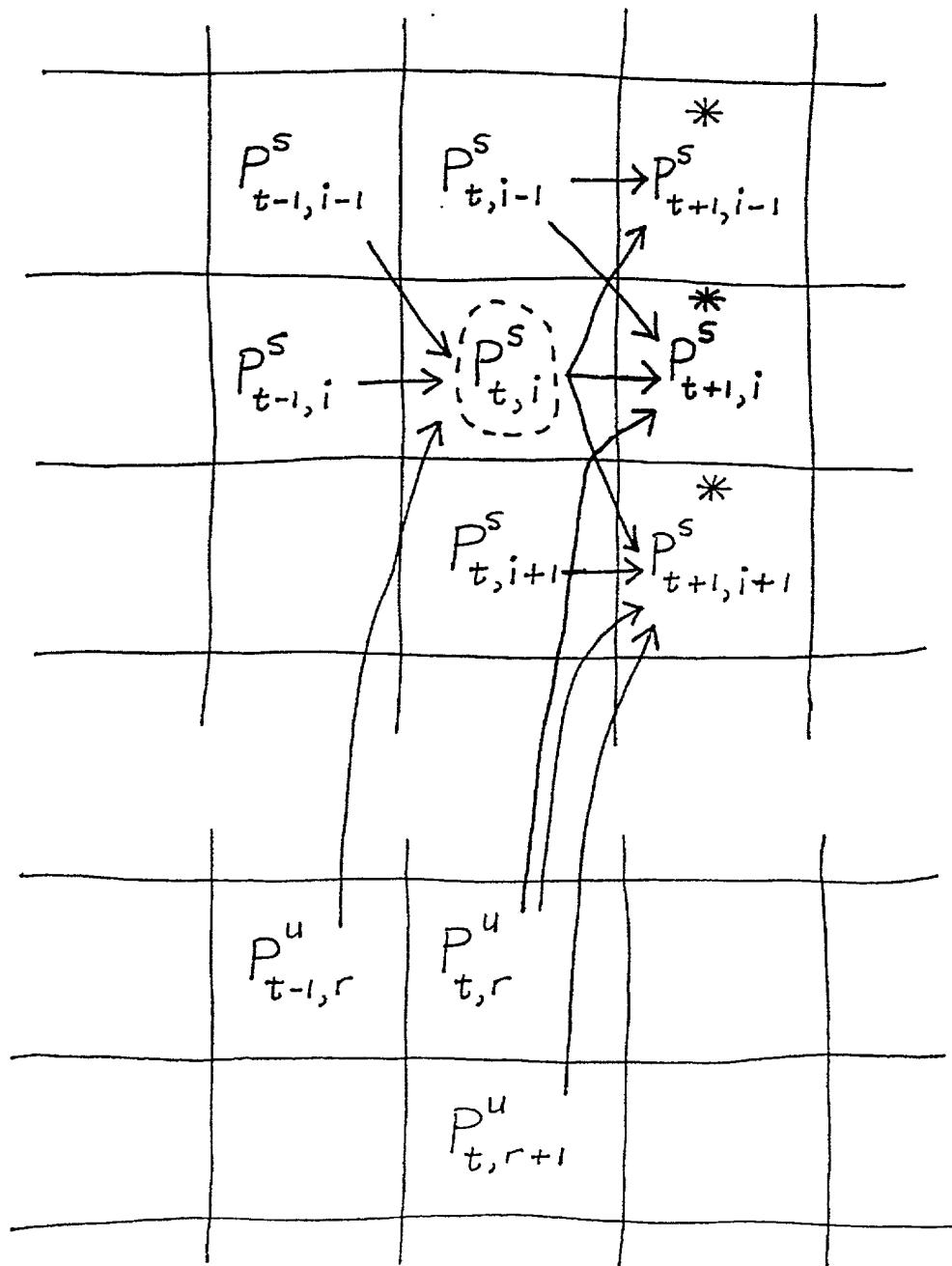
J.H. Kukula, et al.

06816.0036

6/50

FIG. 4C

SSFD



METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

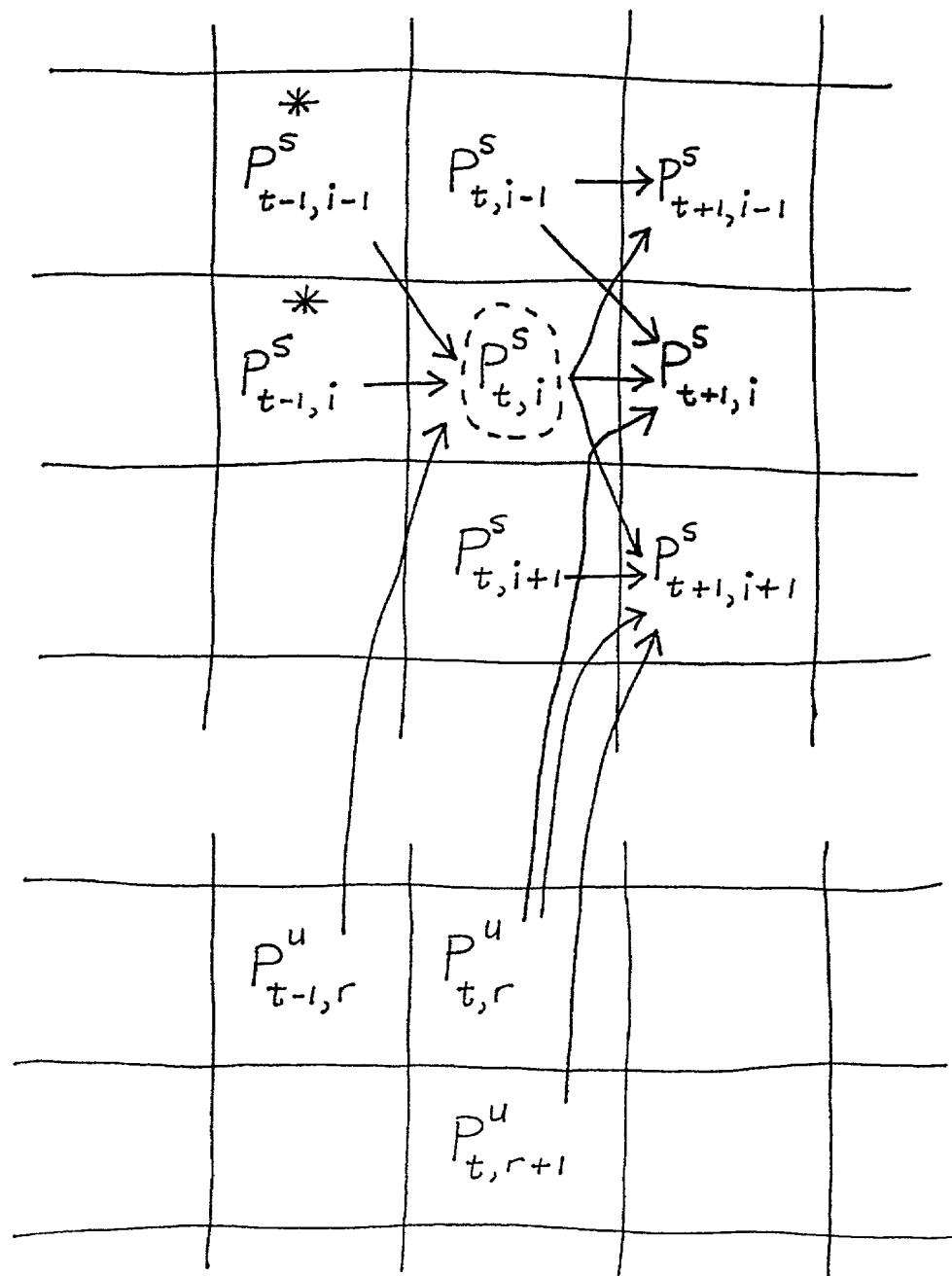
J.H. Kukula, et al.

06816.0036

7/50

FIG. 4D

SSRA



METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

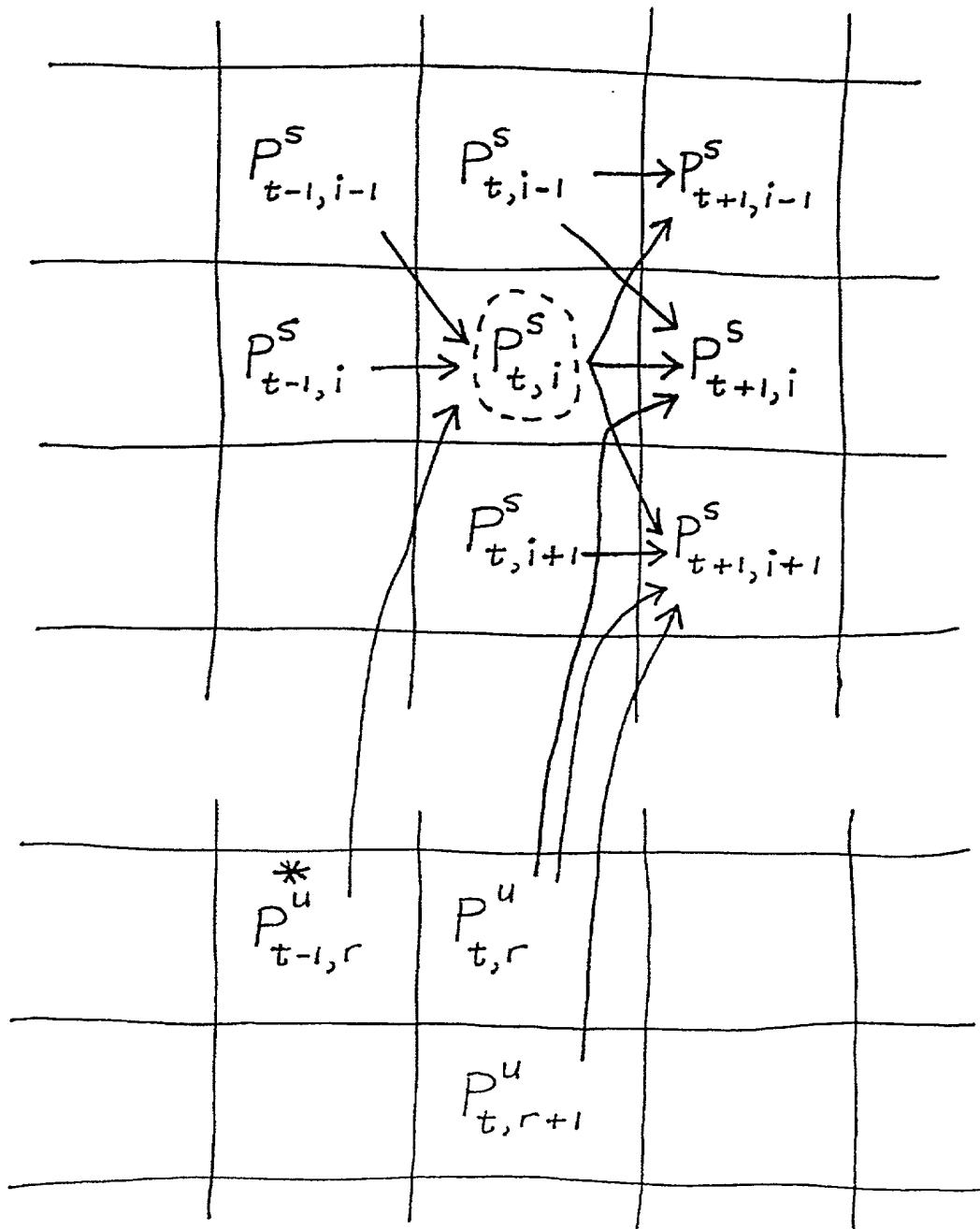
J.H. Kukula, et al.

06816.0036

8/50

FIG. 4E

SURA



METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

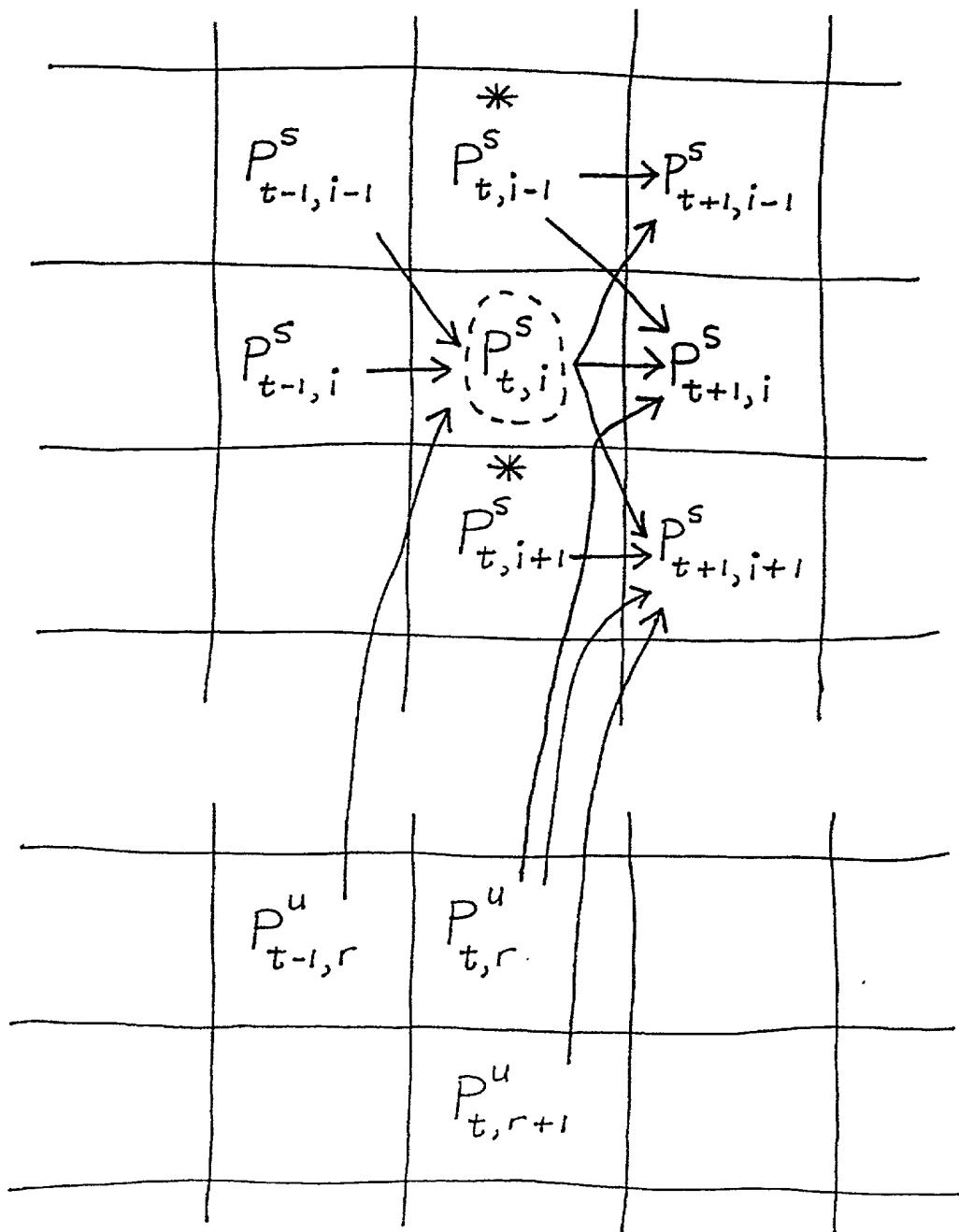
J.H. Kukula, et al.

06816.0036

9/50

FIG. 4F

SSRS



10046220 01.0602

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

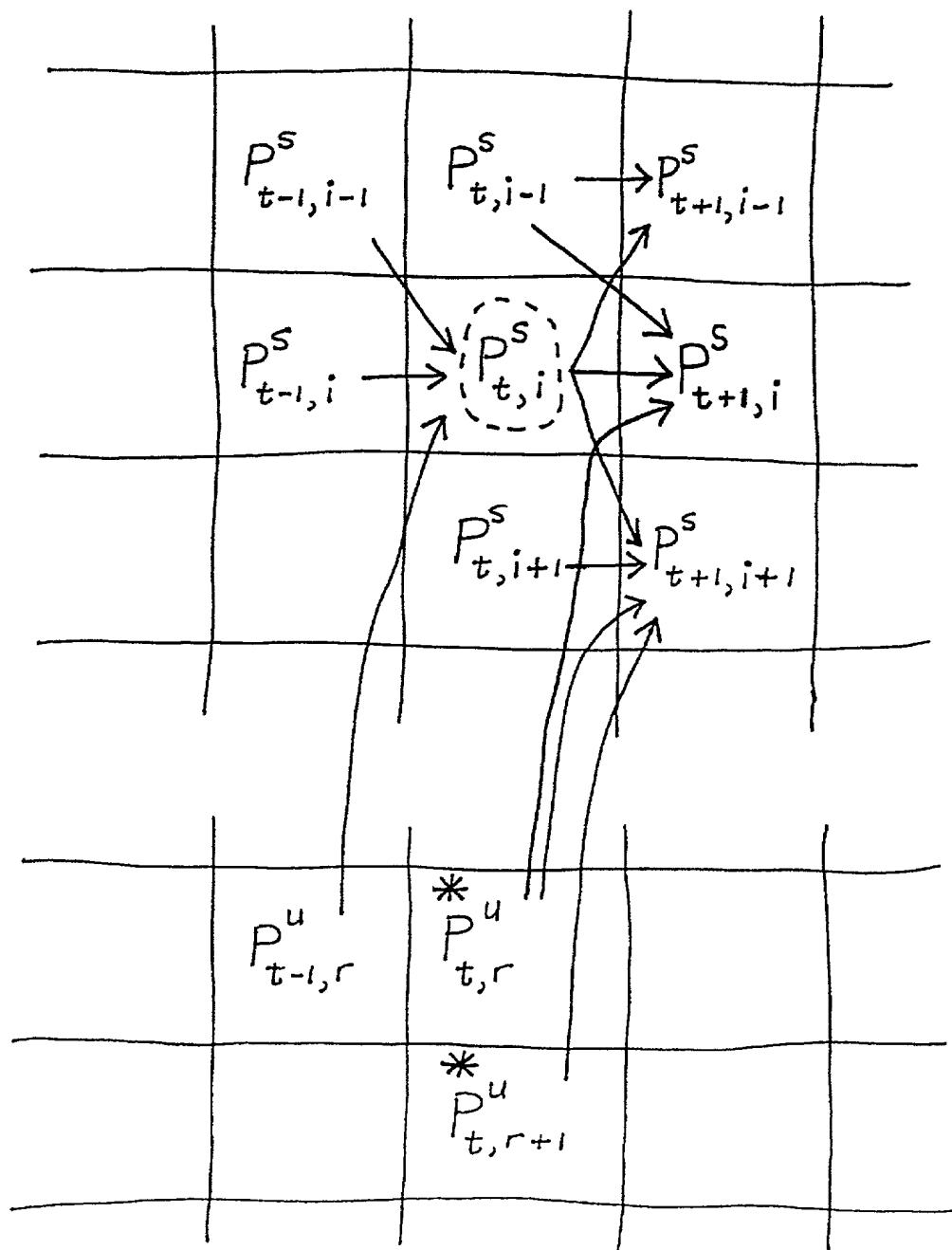
J.H. Kukula, et al.

06816.0036

10/50

FIG. 4G

SURS



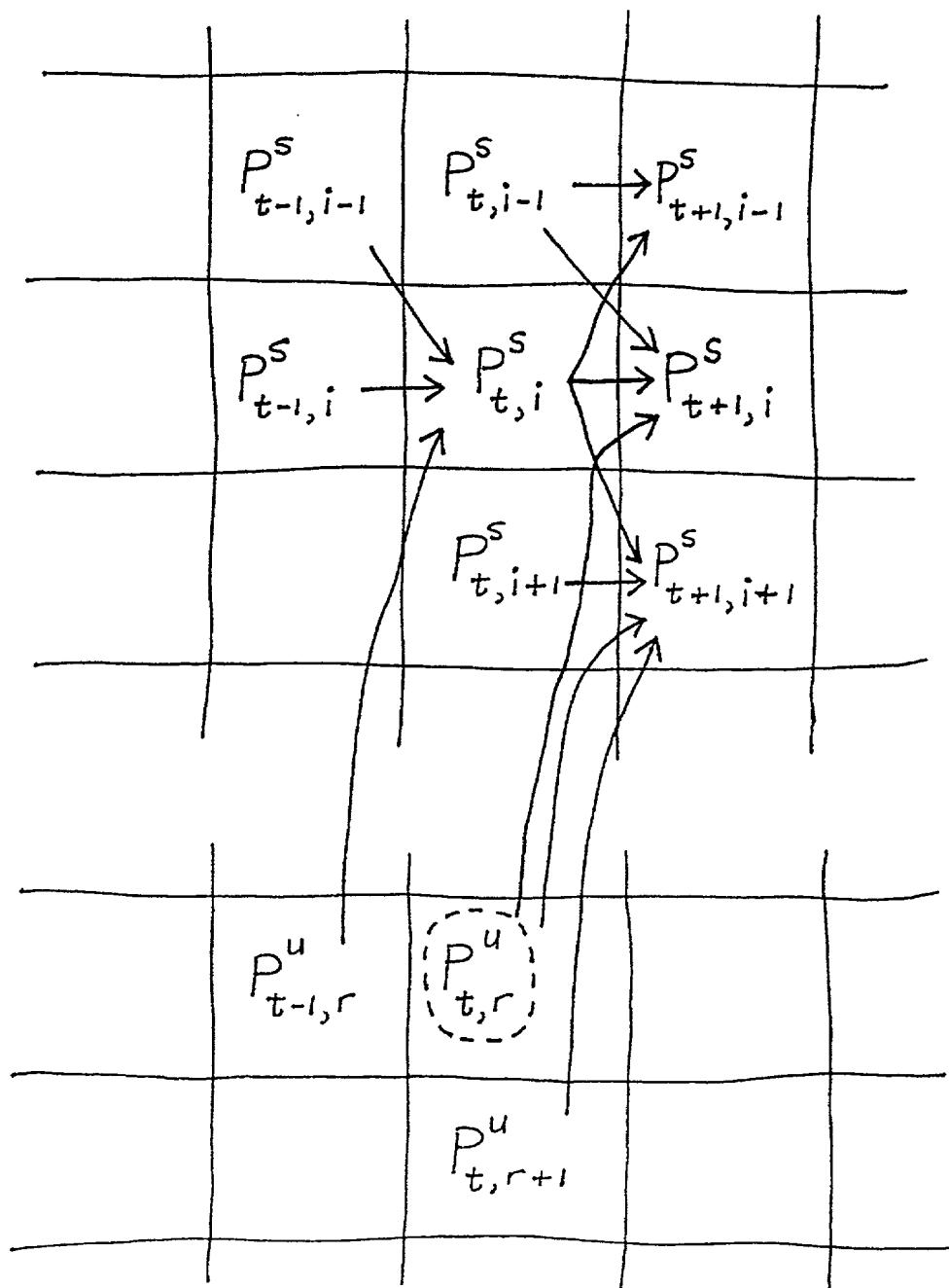
METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

11/50

FIG. 4 H



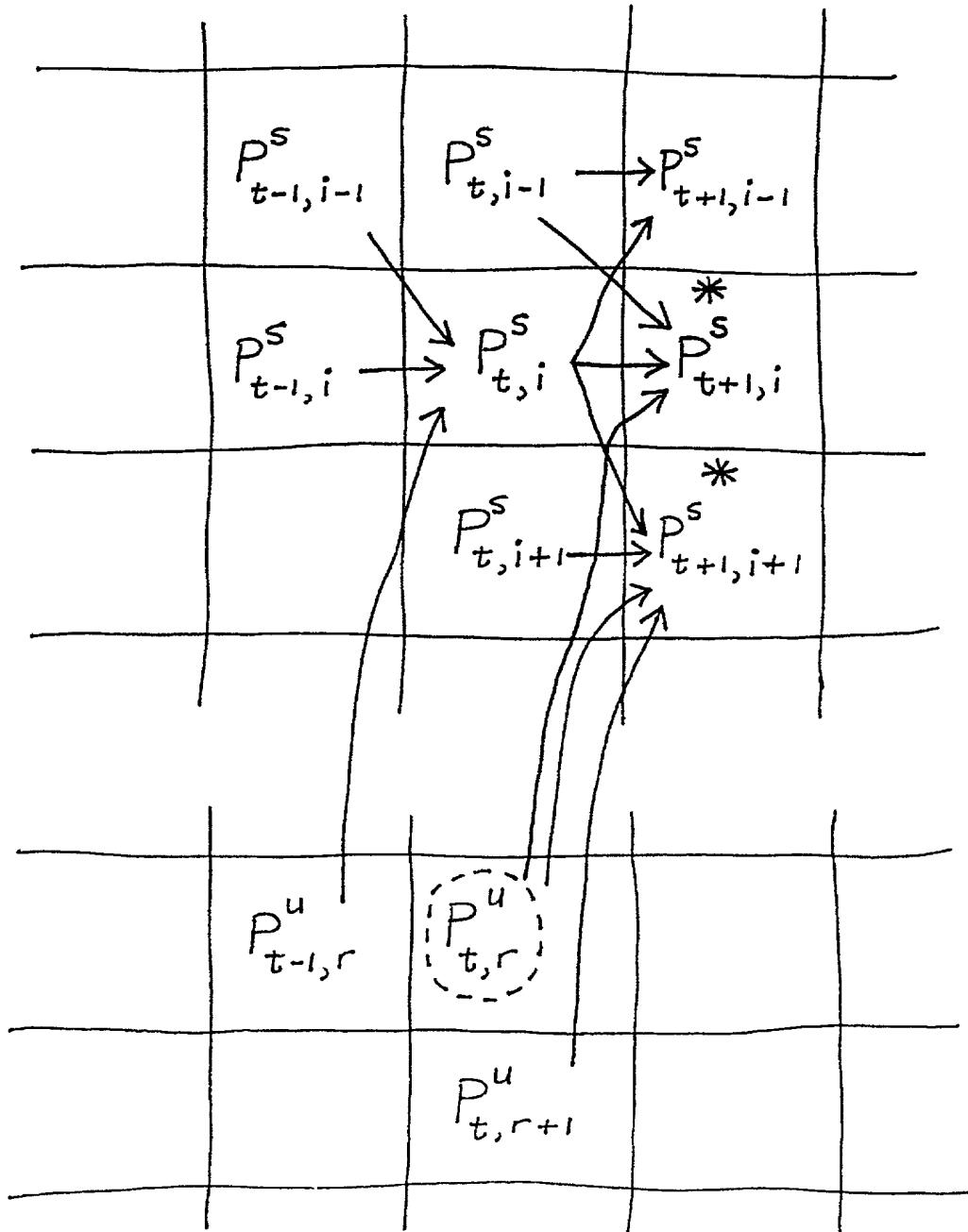
METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

12/50

FIG. 4 I  
USFD



METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

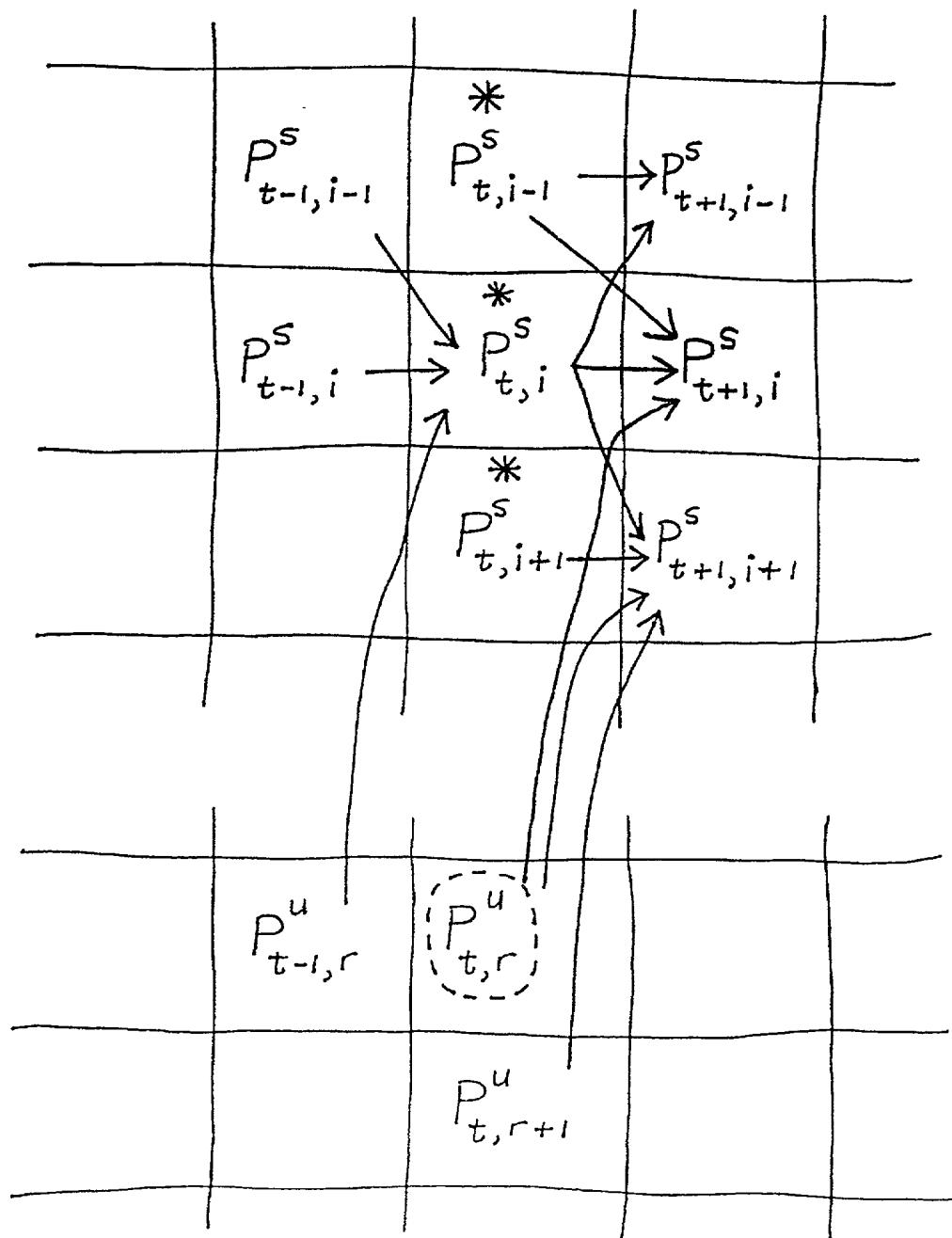
J.H. Kukula, et al.

06816.0036

13/50

FIG. 4J

USRS



METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

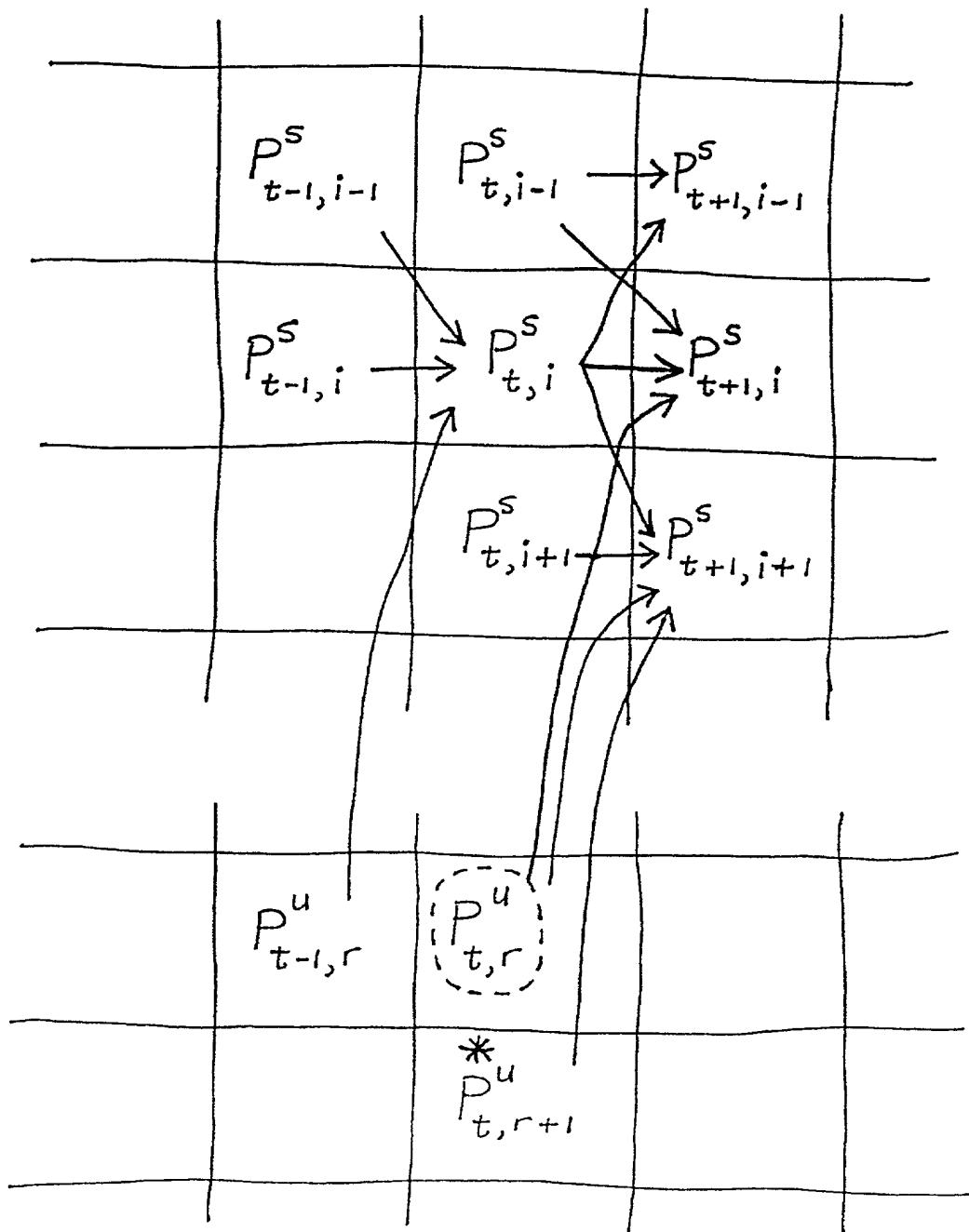
J.H. Kukula, et al.

06816.0036

14/50

FIG. 4K

UURS



METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

15/50

**Figure 5A**

```
1 /* Note that in following declaration "path" is a pseudo data type which is not part
2 of the C language */
3 path approx_path;
4 /* the "approx_path" of the "path" type is structured into the following 3 main
5 subparts:
6     "state_sets" is a matrix organized as follows:
7          $P_{0,1}^S \quad P_{1,1}^S \quad \dots \quad P_{\max\_time,1}^S$ 
8          $P_{0,2}^S \quad P_{1,2}^S \quad \dots \quad P_{\max\_time,2}^S$ 
9         .
10        .
11        .
12          $P_{0,n}^S \quad P_{1,n}^S \quad \dots \quad P_{\max\_time,n}^S$ 
13
14     "input_sets" is a matrix organized as follows:
15          $P_{0,1}^U \quad P_{1,1}^U \quad \dots \quad P_{\max\_time,1}^U$ 
16          $P_{0,2}^U \quad P_{1,2}^U \quad \dots \quad P_{\max\_time,2}^U$ 
17         .
18         .
19         .
20          $P_{0,m}^U \quad P_{1,m}^U \quad \dots \quad P_{\max\_time,m}^U$ 
21
22     "max_time," is the max time value of state_sets and input sets
23 */
```

## Figure 5B

```
1 /* Bidirectional Approximate Reachability Narrowing */
2
3 bidirectional_approx(approx_path)
4 {
5     list rev_comps; /* list of reverse image computations to be performed */
6     list fwd_comps; /* list of forward image computations to be performed */
7
8     /* the approx_path data type is described above */
9
10    Sets of state for max_time,  $P_{\max\_time,1}^S, P_{\max\_time,2}^S, \dots, P_{\max\_time,n}^S$ , are initially shrunk
11    by replacing them with their intersection with  $E_1^S, E_2^S, \dots, E_n^S$ ;
12
13    For each  $P_{\max\_time,i}^S$  shrunk by its intersection with  $E_i^S$  schedule it as the
14    "j" term of any reverse image computations to be performed on its fanin by
15    putting it on rev_comps;
16
17    /* Main loop whereby overapproximate sets, between start and goal states,
18    are shrunken */
19    while (non_empty?( rev_comps) OR non_empty?( fwd_comps))
20    {
21        /* do all rev comps, and all addi rev comps brought up by doing the
22        existing rev comps, on rev comp list, where list is sorted such that
23        rev comps, latest in time, are done first */
24        for each j_term on rev_comps
25        {
26            i_terms = state_fanin(j_term);
27            r_terms = input_fanin(j_term);
```

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

17/50

**Figure 5C**

```
1      for each i_term on i_terms
2      {
3          new_i_term = rev_shrink_2pt2(i_term, j_term);
4          if empty(new_i_term) then return(no_path_exists);
5
6          if new_i_term < i_term then
7          {
8              replace term in approx_path.state_sets,
9              corresponding to i_term, with new_i_term;
10
11         new_j_terms = revs_triggered?(new_i_term);
12
13         put new_j_terms on rev_comps immediately,
14         and re-sort rev_comps such that the j_term's
15         continue to be taken in latest time first order;
16
17         new_fwd_comps =
18         fwds_triggered?(new_i_term);
19
20         put new_fwd_comps on fwd_comps
21         immediately, and re-sort fwd_comps such that
22         its terms continue to be taken in earliest time
23         first order;
24     } /* END if new_i_term < i_term */
25 } /* END for each i_term on i_terms */
```

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

18/50

**Figure 5D**

```
1      for each r_term on r_terms
2      {
3          new_r_term = rev_shrink_2pt3(r_term, j_term);
4          if empty(new_r_term) then return(no_path_exists);
5
6          if new_r_term < r_term then
7          {
8              replace term in approx_path.input_sets,
9              corresponding to r_term, with new_r_term;
10
11             new_j_terms = revs_triggered?(new_r_term);
12
13             put new_j_terms on rev_comps immediately,
14             and re-sort rev_comps such that the j_term's
15             continue to be taken in latest time first order;
16
17             new_fwd_comps =
18             fwds_triggered?(new_r_term);
19
20             put new_fwd_comps on fwd_comps
21             immediately, and re-sort fwd_comps such that
22             its terms continue to be taken in earliest time
23             first order;
24         } /* END if new_r_term < r_term */
25     } /* END for each r_term on r_terms */
26 } /* END for each j_term on rev_comps */
```

**METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION**

J.H. Kukula, et al.

06816.0036

**20/50**

## **Figure 6A**

```

1
2
3
4 /* All calls or process spawnings in the following pseudocode are by value,
5 meaning that the called routine or spawned process gets its own copy of the
6 passed parameters. */
7
8 /* DECLARATIONS FOLLOW (note that certain types are "pseudo" data types not
9 part of the C language) */
10
11 int lwr_prio;
12 int max_prio;
13
14 state initial_state;
15 list error_states;
16 list actual_path;
17 hash previously_found_states; /* "hash" is a pseudo data type which creates a
18 hash table; previously_found_states is a global hash table where all states
19 generated, beyond initial_state, are kept track of */
20 path approx_path;
21 /* the "approx_path" of the "path" type is structured into the following 3 main
22 subparts:
23     "state_sets" is a matrix organized as follows:
24          $P_{0,1}^S \quad P_{1,1}^S \quad \dots \quad P_{\max\_time,1}^S$ 
25          $P_{0,2}^S \quad P_{1,2}^S \quad \dots \quad P_{\max\_time,2}^S$ 
26         .
27         .
28         .
29          $P_{0,n}^S \quad P_{1,n}^S \quad \dots \quad P_{\max\_time,n}^S$ 
30
31     "input_sets" is a matrix organized as follows:
32          $P_{0,1}^U \quad P_{1,1}^U \quad \dots \quad P_{\max\_time,1}^U$ 
33          $P_{0,2}^U \quad P_{1,2}^U \quad \dots \quad P_{\max\_time,2}^U$ 
34         .
35         .
36         .
37          $P_{0,m}^U \quad P_{1,m}^U \quad \dots \quad P_{\max\_time,m}^U$ 
38
39     "max_time," is the max time value of state_sets and input sets*/

```

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

21/50

**Figure 6B**

```
1 /* INITIALIZATIONS FOLLOW: */  
2  
3 /* Note that priority decreases as the priority number increases */  
4 lwr_prio = 1;  
5 max_prio = 1;  
6  
7 initial_state =  $s_{0,1}, s_{0,2} \dots s_{0,n}$  ;  
8 error_states =  $E_1^S, E_2^S, \dots E_n^S$  ;  
9 actual_path = ( (( $s_{0,1}, s_{0,2} \dots s_{0,n}$ ), NULL_input) ); /* Note that initial state of actual  
10 path is paired with NULL_input to symbolize that primary input combination to be  
11 applied for getting to next state along an error path has not been found yet. */
```

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

22/50

## Figure 6C

```
1 /* Initial Process */
2
3 approx_path is initialized as follows:
4     state_sets (each set  $P_{0,t}^S$  accepts only its corresponding portion of initial
5     state  $s_{0,1}, s_{0,2} \dots s_{0,n}$ ):
6          $P_{0,1}^S$ 
7          $P_{0,2}^S$ 
8         .
9         .
10        .
11         $P_{0,n}^S$ 
12
13     input_sets (each set  $P_{0,r}^U$  accepts any combination of inputs applied to it):
14          $P_{0,1}^U$ 
15          $P_{0,2}^U$ 
16         .
17         .
18         .
19          $P_{0,m}^U$ 
20
21     max_time is 0
22
23 if each partition of initial_state has a non-null intersection with the corresponding
24 partition of error_states
25     then return initial_state as being a path to an error and end verification;
26
27     else spawn_process( max_prio; forward_approx(approx_path, actual_path)
28 )
```

**METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION**

J.H. Kukula, et al.

06816.0036

**23/50**

## **Figure 6D**

```

1  /* Forward Approximate Reachability Process */
2
3  forward_approx(approx_path, actual_path)
4
5      Determine  $P_{t,1}^S, P_{t,2}^S, \dots, P_{t,n}^S$  from  $P_{t-1,1}^S, P_{t-1,2}^S, \dots, P_{t-1,n}^S$ , where each  $P_{t,i}^S$  is
6      determined according to the following equation:
7
8          
$$P_{t,i}^S(s_{t,i}) = \exists s_{t-1,a_1}, \exists s_{t-1,a_2}, \dots, \exists s_{t-1,a_q}, \exists u_i$$

9          
$$[P_{t-1,a_1}^S(s_{t-1,a_1}), P_{t-1,a_2}^S(s_{t-1,a_2}), \dots, P_{t-1,a_q}^S(s_{t-1,a_q}) \wedge$$

10         
$$T_t(s_{t-1,a_1}, s_{t-1,a_2}, \dots, s_{t-1,a_q}, u_i, s_{t,i})]$$

11
12
13
14  aug_approx_path = approx_path with the additional time step values of
15   $P_{t,1}^S, P_{t,2}^S, \dots, P_{t,n}^S$  added to state_sets, and max_time incremented by 1;
16
17  spawn_process( local_prio () + lwr_prio; forward_approx(aug_approx_path,
18  actual_path) );
19      /* local_prio () returns priority level of process in which local_prio is
20      being executed */
21
22      /* start another forward_approx, but at a lwr level of priority */
23
24  if intersection of each member of  $P_{t,1}^S, P_{t,2}^S, \dots, P_{t,n}^S$  with its corresponding
25  member of  $E_1^S, E_2^S, \dots, E_n^S$  is non-null then spawn_process(max_prio,
26  bidirectional_approx(aug_approx_path, actual_path));

```

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

24/50

## Figure 6E

```
1 /* Bidirectional Approximate Reachability Narrowing */
2
3 bidirectional_approx(approx_path, actual_path)
4 {
5     list rev_comps; /* list of reverse image computations to be performed */
6     list fwd_comps; /* list of forward image computations to be performed */
7
8     /* the approx_path data type is described above */
9
10    Sets of state for max_time,  $P_{\max\_time,1}^S, P_{\max\_time,2}^S, \dots, P_{\max\_time,n}^S$ , are shrunk by
11    replacing them with their intersection with  $E_1^S, E_2^S, \dots, E_n^S$ ;
12
13    For each  $P_{\max\_time,i}^S$  shrunken by its intersection with  $E_i^S$  schedule it as the
14    "j" term of any reverse image computations to be performed on its fanin by
15    putting it on rev_comps;
16
17    /* Main loop whereby overapproximate sets, between start and goal states,
18    are shrunken */
19    while (non_empty?( rev_comps) OR non_empty?( fwd_comps))
20    {
21        /* do all rev comps, and all addi rev comps brought up by doing the
22        existing rev comps, on rev comp list, where list is sorted such that
23        rev comps, latest in time, are done first */
24        for each j_term on rev_comps
25        {
26            i_terms = state_fanin(j_term);
27            r_terms = input_fanin(j_term);
```

## METHOD AND APPARATUS FOR FORMALLY CONSTRAINING RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

25/50

**Figure 6F**

```

1   for each i_term on i_terms
2   {
3       new_i_term = rev_shrink_2pt2(i_term, j_term);
4       if empty(new_i_term) then return(no_path_exists);
5
6       if new_i_term < i_term then
7       {
8           replace term in approx_path.state_sets,
9           corresponding to i_term, with new_i_term;
10
11      new_j_terms = revs_triggered?(new_i_term);
12
13      put new_j_terms on rev_comps immediately,
14      and re-sort rev_comps such that the j_term's
15      continue to be taken in latest time first order;
16
17      new_fwd_comps =
18      fwds_triggered?(new_i_term);
19
20      put new_fwd_comps on fwd_comps
21      immediately, and re-sort fwd_comps such that
22      its terms continue to be taken in earliest time
23      first order;
24      } /* END if new_i_term < i_term */
25  } /* END for each i_term on i_terms */

```

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

26/50

**Figure 6G**

```
1   for each r_term on r_terms
2   {
3       new_r_term = rev_shrink_2pt3(r_term, j_term);
4       if empty(new_r_term) then return(no_path_exists);
5
6       if new_r_term < r_term then
7       {
8           replace term in approx_path.input_sets,
9           corresponding to r_term, with new_r_term;
10
11      new_j_terms = revs_triggered?(new_r_term);
12
13      put new_j_terms on rev_comps immediately,
14      and re-sort rev_comps such that the j_term's
15      continue to be taken in latest time first order;
16
17      new_fwd_comps =
18      fwds_triggered?(new_r_term);
19
20      put new_fwd_comps on fwd_comps
21      immediately, and re-sort fwd_comps such that
22      its terms continue to be taken in earliest time
23      first order;
24  } /* END if new_r_term < r_term */
25  /* END for each r_term on r_terms */
```

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

27/50

**Figure 6H**

```
1  /* do all forward comps, and all addi fwd comps brought up by doing
2   the existing fwd comps, on fwd comp list, where list is sorted such
3   that fwd comps, earliest in time, are done first */
4   for each i_term on fwd_comps
5   {
6       new_i_term = fwd_shrink_2pt1(i_term);
7       if empty(new_i_term) then return(no_path_exists);
8
9       if new_i_term < i_term then
10      {
11          replace term in approx_path.state_sets, corresponding
12          to i_term, with new_i_term;
13
14          new_j_terms = revs_triggered?(new_i_term);
15
16          put new_j_terms on rev_comps immediately, and re-
17          sort rev_comps such that the j_term's continue to be
18          taken in latest time first order;
19
20          new_fwd_comps = fwds_triggered?(new_i_term);
21
22          put new_fwd_comps on fwd_comps immediately, and re-
23          sort fwd_comps such that its terms continue to be
24          taken in earliest time first order;
25      } /* END if new_i_term < i_term */
26  } /* END for each i_term on fwd_comps */
27
28 } /* END while */
29
30 random_seed = random();
31 spawn_process( max_prio; simulate(approx_path, actual_path,
32 random_seed));
33
34 } /* END bidirectional_approx */
```

## Figure 6I

```
1 /* Simulation Process
2 Want to simulate one step from end of actual_path, where end of actual_path is
3 also the only state contained in  $P_{0,1}^S, P_{0,2}^S, \dots, P_{0,n}^S$  of approx_path. */
4 simulate(approx_path, actual_path)
5 {
6
7     /* spawn another simulation process where the random_seed has a
8      different value to ensure a different random vector of inputs to try */
9     spawn_process( local_prio() + lwr_prio, simulate(approx_path,
10                  actual_path))
11
12     end_of_path = get_end_of_path( actual_path )
13
14     /* each call to random_valid_input returns a different randomly generated
15      set of inputs which are a member of  $P_{0,1}^U, P_{0,2}^U, \dots, P_{0,m}^U$  */
16     input_vector = random_valid_input(approx_path);
17
18     /* simulate  $FSM_{verify}$  for one step, from the time 0 state of approx_path, with
19      the randomly generated set of inputs of input_vector */
20     next_state = one_step_fsm_verify(end_of_path, input_vector);
21
22     new_actual_path = replace the “NULL_input” paired with end_of_path in
23     actual_path with input_vector;
24
25     new_actual_path = concatenate (next_state, NULL_input) pair to end of
26     existing list assigned to new_actual_path;
27
28     if next_state is contained in  $E_1^S, E_2^S, \dots, E_n^S$  then end entire search and return
29     new_actual_path to user as a concrete path from initial state to an error
30     state, otherwise continue;
```

## Figure 6J

```
1   if ( (next_state contained in  $P_{1,1}^S, P_{1,2}^S, \dots, P_{1,n}^S$ ) && not(next_state contained in
2     previously_found_states) )
3   {
4     /* add next_state to global hash table, previously_found_states, so
5       that next_state will not be pursued by another process */
6     add_state_to_table(next_state, previously_found_states);
7
8     new_approx_path = only has state_sets for  $P_{0,1}^S, P_{0,2}^S, \dots, P_{0,n}^S$ , which only
9       contain next_state; only has input_sets  $P_{0,1}^U, P_{0,2}^U, \dots, P_{0,m}^U$ , which can
10      accept any input combination; and max_time is set to zero;
11
12     spawn_process( local_prio(), forward_approx(new_approx_path,
13       new_actual_path) );
14   }
15
16 } /* END simulate */
17
18
19 /* local_prio () returns priority level of process in which local_prio is being executed
20 */
21 local_prio()
22 {
23   return priority level of process in which local_prio just called
24 }
```

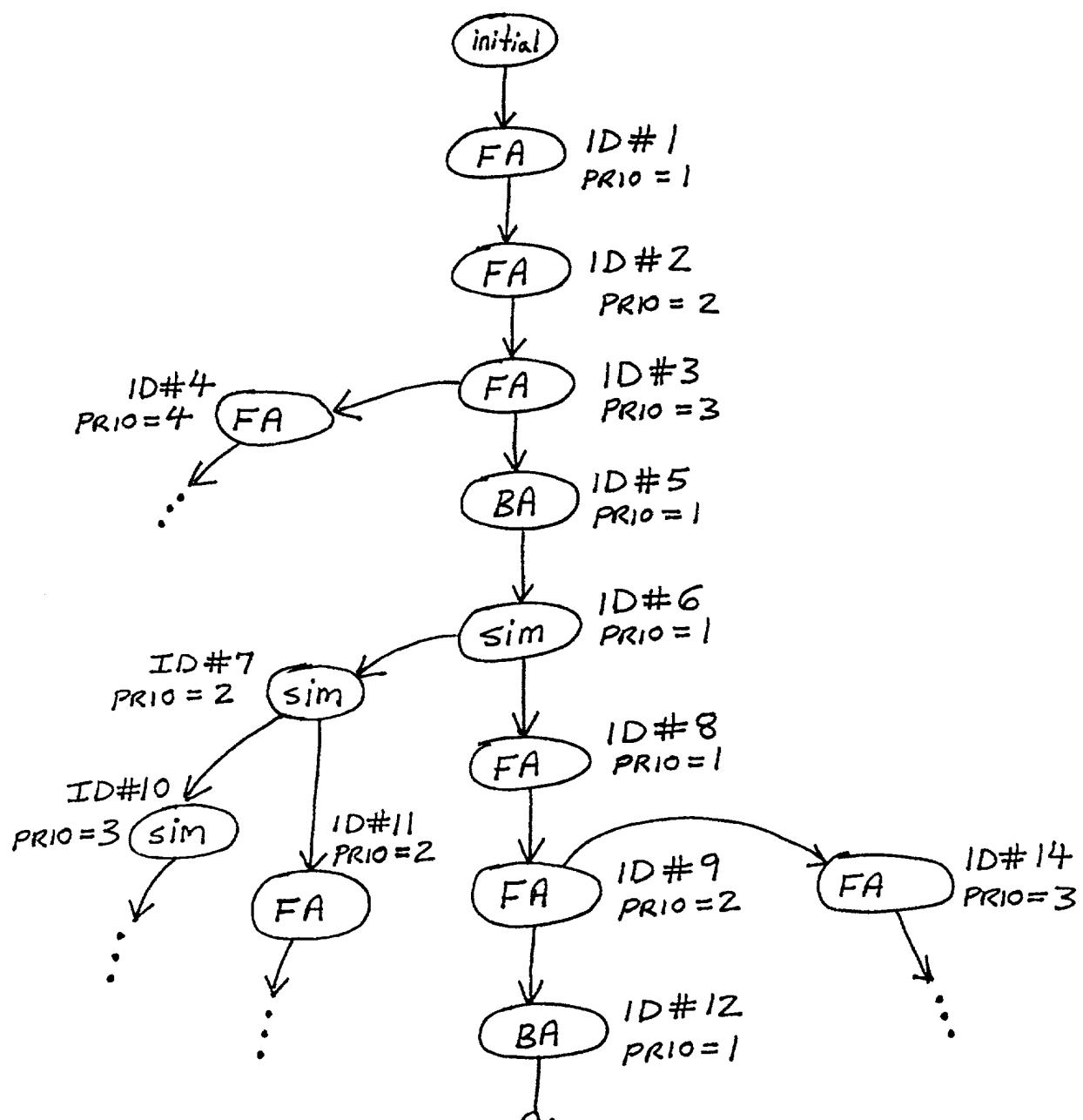
METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

30/50

Figure 7A



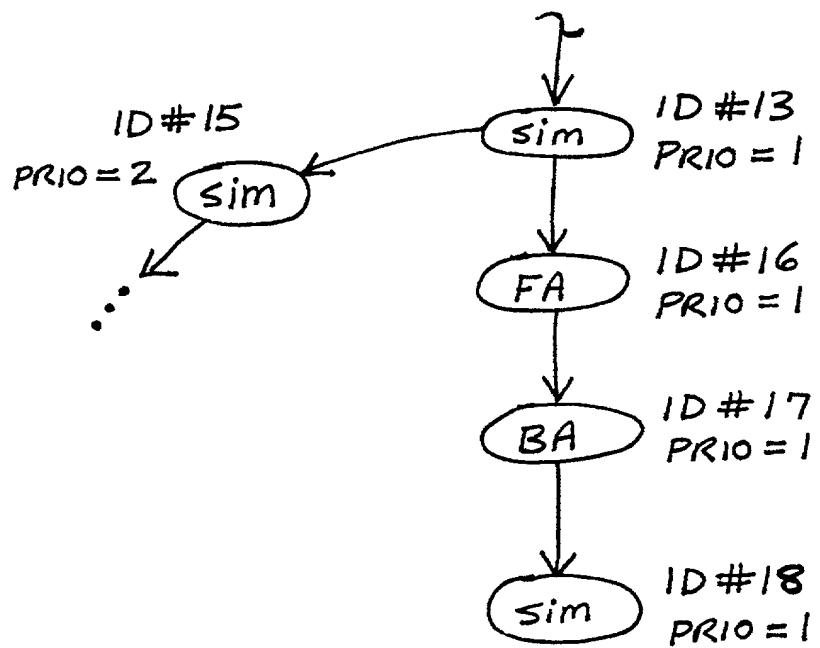
METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

31/50

Figure 7B



METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

32/50

## Figure 8A

1   Initial State

2

3   The following values are determined by initializations:

4       actual\_path = ( (  $s_{0,1}, s_{0,2}, s_{0,3}$  ) )

5

6       approx\_path =  
7           state\_sets

8               $P_{0,1}^S$

9               $P_{0,2}^S$

10              $P_{0,3}^S$

11        input\_sets

12               $P_{0,1}^U$

13               $P_{0,2}^U$

14

15       max\_time = 0

16

17   Notes:

18       state\_sets accept only initial state  $s_{0,1}, s_{0,2}, s_{0,3}$ .

19

20       input\_sets accept any input combination

21

22       spawn Forward Approximation, Process ID#1

## Figure 8B

1 Forward Approximation: Process ID#1

2  
3 The following values are unchanged by forward\_approx:  
4     actual\_path = ( (s<sub>0,1</sub>, s<sub>0,2</sub>, s<sub>0,3</sub>) )  
5  
6     local priority = 1  
7

8 The following values are after having been changed by forward\_approx:

9     approx\_path =  
10         state\_sets  
11             P<sub>0,1</sub><sup>S</sup>     P<sub>1,1</sub><sup>S</sup>  
12             P<sub>0,2</sub><sup>S</sup>     P<sub>1,2</sub><sup>S</sup>  
13             P<sub>0,3</sub><sup>S</sup>     P<sub>1,3</sub><sup>S</sup>  
14         input\_sets  
15             P<sub>0,1</sub><sup>U</sup>     P<sub>1,1</sub><sup>U</sup>  
16             P<sub>0,2</sub><sup>U</sup>     P<sub>1,2</sub><sup>U</sup>  
17  
18         max\_time = 1  
19

20 Notes:

21     state\_sets of time 1 is an overapproximation of the reachable states

22  
23     input\_sets at times 0 and 1 accept any input combination

24  
25     spawn Forward Approximation, Process ID#2

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

34/50

## Figure 8C

1 Forward Approximation: Process ID#2

2  
3 The following values are unchanged by forward\_approx:  
4     actual\_path = ( (s<sub>0,1</sub>, s<sub>0,2</sub>, s<sub>0,3</sub>) )

5  
6     local priority = 2  
7

8 The following values are after having been changed by forward\_approx:

9     approx\_path =  
10       state\_sets

$$\begin{array}{ccc} P_{0,1}^S & P_{1,1}^S & P_{2,1}^S \\ P_{0,2}^S & P_{1,2}^S & P_{2,2}^S \\ P_{0,3}^S & P_{1,3}^S & P_{2,3}^S \end{array}$$

11       input\_sets

$$\begin{array}{ccc} P_{0,1}^U & P_{1,1}^U & P_{2,1}^U \\ P_{0,2}^U & P_{1,2}^U & P_{2,2}^U \end{array}$$

12  
13  
14  
15  
16  
17  
18     max\_time = 2  
19

20 Notes:

21     state\_sets of times 0-1 are unchanged; state\_sets of time 2 are an  
22     overapproximation of the states reachable from the state\_sets of time 1.

23  
24     input\_sets at times 0-2 accept any input combination.

25  
26     spawn Forward Approximation, Process ID#3

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

35/50

## Figure 8D

1 Forward Approximation: Process ID#3

2  
3 The following values are unchanged by forward\_approx:

4      actual\_path = ( (s<sub>0,1</sub>, s<sub>0,2</sub>, s<sub>0,3</sub>) )

5  
6      local priority = 3

7  
8 The following values are after having been changed by forward\_approx:

9      approx\_path =  
10        state\_sets

11            P<sub>0,1</sub><sup>S</sup>    P<sub>1,1</sub><sup>S</sup>    P<sub>2,1</sub><sup>S</sup>    P<sub>3,1</sub><sup>S</sup>

12            P<sub>0,2</sub><sup>S</sup>    P<sub>1,2</sub><sup>S</sup>    P<sub>2,2</sub><sup>S</sup>    P<sub>3,2</sub><sup>S</sup>

13            P<sub>0,3</sub><sup>S</sup>    P<sub>1,3</sub><sup>S</sup>    P<sub>2,3</sub><sup>S</sup>    P<sub>3,3</sub><sup>S</sup>

14        input\_sets

15            P<sub>0,1</sub><sup>U</sup>    P<sub>1,1</sub><sup>U</sup>    P<sub>2,1</sub><sup>U</sup>    P<sub>3,1</sub><sup>U</sup>

16            P<sub>0,2</sub><sup>U</sup>    P<sub>1,2</sub><sup>U</sup>    P<sub>2,2</sub><sup>U</sup>    P<sub>3,2</sub><sup>U</sup>

17  
18        max\_time = 3

20 Notes:

21        state\_sets of times 0-2 are unchanged; state\_sets of time 3 are an  
22        overapproximation the states reachable from the state\_sets of time 2.

24        input\_sets at times 0-3 accept any input combination.

26        spawn Forward Approximation, Process ID#4 (not shown)

28        spawn Bidirectional Approximation, Process ID#5

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

36/50

## Figure 8E

1    Bidirectional Approximation: Process ID#5

2    The following values are unchanged by bidirectional\_approx:

3        actual\_path = ( (s<sub>0,1</sub>, s<sub>0,2</sub>, s<sub>0,3</sub>) )

4        local priority = 1

5    The following values are after having been changed by bidirectional\_approx:

6        approx\_path =  
7              state\_sets

8              P<sub>0,1</sub><sup>S</sup>    P<sub>1,1</sub><sup>S</sup>    P<sub>2,1</sub><sup>S</sup>    P<sub>3,1</sub><sup>S</sup>

9              P<sub>0,2</sub><sup>S</sup>    P<sub>1,2</sub><sup>S</sup>    P<sub>2,2</sub><sup>S</sup>    P<sub>3,2</sub><sup>S</sup>

10             P<sub>0,3</sub><sup>S</sup>    P<sub>1,3</sub><sup>S</sup>    P<sub>2,3</sub><sup>S</sup>    P<sub>3,3</sub><sup>S</sup>

11             input\_sets

12             P<sub>0,1</sub><sup>U</sup>    P<sub>1,1</sub><sup>U</sup>    P<sub>2,1</sub><sup>U</sup>    P<sub>3,1</sub><sup>U</sup>

13             P<sub>0,2</sub><sup>U</sup>    P<sub>1,2</sub><sup>U</sup>    P<sub>2,2</sub><sup>U</sup>    P<sub>3,2</sub><sup>U</sup>

14             max\_time = 3

15    Notes:

16        state\_sets of time 3 are narrowed to their intersection with E<sub>1</sub><sup>S</sup>, E<sub>2</sub><sup>S</sup>, ... E<sub>n</sub><sup>S</sup>.

17        state\_sets of times 1-2 may be narrowed by forward image or reverse  
18        image computations.

19        input\_sets of times 0-2 may be narrowed by reverse image computations.

20        spawn Simulation, Process ID#6

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

37/50

## Figure 8F

1   Simulation: Process ID#6

2   proceeds according to the following major steps:

3       begins with following passed parameters and values:

4           local priority = 1

5

6

7       actual\_path = ( (  $s_{0,1}, s_{0,2}, s_{0,3}$  ) )

8

9       approx\_path =  
10          state\_sets

11             $P_{0,1}^S$      $P_{1,1}^S$      $P_{2,1}^S$      $P_{3,1}^S$

12             $P_{0,2}^S$      $P_{1,2}^S$      $P_{2,2}^S$      $P_{3,2}^S$

13             $P_{0,3}^S$      $P_{1,3}^S$      $P_{2,3}^S$      $P_{3,3}^S$

14          input\_sets

15             $P_{0,1}^U$      $P_{1,1}^U$      $P_{2,1}^U$      $P_{3,1}^U$

16             $P_{0,2}^U$      $P_{1,2}^U$      $P_{2,2}^U$      $P_{3,2}^U$

17

18       max\_time = 3

19

20       spawn a Simulation, with the old parameters, Process ID#7 (not shown in  
21       Figure 8)

22

23       A one step simulation of  $FSM_{verify}$  is performed to produce a next\_state. A

24       next\_state  $s_{1,1}, s_{1,2}, s_{1,3}$  is found by using a randomly selected input

25       combination contained in  $P_{0,1}^U, P_{0,2}^U$  in combination with  $s_{0,1}, s_{0,2}, s_{0,3}$ .

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

38/50

## Figure 8G

From the parameters passed to simulation, the following new parameters are determined:

```
new_actual_path = ( (s0,1, s0,2, s0,3), (s1,1, s1,2, s1,3) )
new_approx_path =
state_sets
    P0,1S
    P0,2S
    P0,3S
input_sets
    P0,1U
    P0,2U
max_time = 0
```

Notes:

The column of state\_sets only contains  $s_{1,1}, s_{1,2}, s_{1,3}$ .

The column of input\_sets accepts any input combination.

spawn a Forward Approximation, with the new parameters, Process ID#8

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

,06816.0036

**39/50**

## Figure 8H

1   Forward Approximation: Process ID#8

2  
3   The following values are unchanged by forward\_approx:

4       actual\_path = ( (s<sub>0,1</sub>, s<sub>0,2</sub>, s<sub>0,3</sub>), (s<sub>1,1</sub>, s<sub>1,2</sub>, s<sub>1,3</sub>) )

5  
6       local priority = 1

7  
8   The following values are after having been changed by forward\_approx:

9       approx\_path =  
10           state\_sets

$$P_{0,1}^S \quad P_{1,1}^S$$

$$P_{0,2}^S \quad P_{1,2}^S$$

$$P_{0,3}^S \quad P_{1,3}^S$$

11           input\_sets

$$P_{0,1}^U \quad P_{1,1}^U$$

$$P_{0,2}^U \quad P_{1,2}^U$$

12  
13  
14  
15  
16  
17  
18       max\_time = 1

19  
20   Notes:

21       spawn Forward Approximation, Process ID#9

## Figure 8I

1 Forward Approximation: Process ID#9

2  
3 The following values are unchanged by forward\_approx:

4      actual\_path = ( (s<sub>0,1</sub>, s<sub>0,2</sub>, s<sub>0,3</sub>), (s<sub>1,1</sub>, s<sub>1,2</sub>, s<sub>1,3</sub>) )

5  
6      local priority = 2

7  
8 The following values are after having been changed by forward\_approx:

9      approx\_path =  
10        state\_sets

$$P_{0,1}^S \quad P_{1,1}^S \quad P_{2,1}^S$$

$$P_{0,2}^S \quad P_{1,2}^S \quad P_{2,2}^S$$

$$P_{0,3}^S \quad P_{1,3}^S \quad P_{2,3}^S$$

11        input\_sets

$$P_{0,1}^U \quad P_{1,1}^U \quad P_{2,1}^U$$

$$P_{0,2}^U \quad P_{1,2}^U \quad P_{2,2}^U$$

12  
13  
14  
15  
16  
17  
18        max\_time = 2

19  
20 Notes:

21        spawn Forward Approximation, Process ID#14 (not shown)

22  
23        spawn Bidirectional Approximation, Process ID#12

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

41/50

## Figure 8J

1    Bidirectional Approximation: Process ID#12

2    The following values are unchanged by bidirectional\_approx:

3       actual\_path = ( (  $s_{0,1}, s_{0,2}, s_{0,3}$  ), (  $s_{1,1}, s_{1,2}, s_{1,3}$  ) )

4       local priority = 1

5    The following values are after having been changed by bidirectional\_approx:

6       approx\_path =

7          state\_sets

8              $P_{0,1}^S$      $P_{1,1}^S$      $P_{2,1}^S$

9              $P_{0,2}^S$      $P_{1,2}^S$      $P_{2,2}^S$

10              $P_{0,3}^S$      $P_{1,3}^S$      $P_{2,3}^S$

11             input\_sets

12                 $P_{0,1}^U$      $P_{1,1}^U$      $P_{2,1}^U$

13                 $P_{0,2}^U$      $P_{1,2}^U$      $P_{2,2}^U$

14                max\_time = 2

15    Notes:

16    state\_sets of time 2 are narrowed to their intersection with  $E_1^S, E_2^S, \dots, E_n^S$ .

17    state\_sets of time 1 may be narrowed by forward image or reverse image computations.

18    input\_sets of times 0-1 may be narrowed by reverse image computations.

19    spawn Simulation, Process ID#13

## Figure 8K

1   Simulation: Process ID#13

2   proceeds according to the following major steps:

3       begins with following passed parameters and values:

4           local priority = 1

5           actual\_path = ( (  $s_{0,1}, s_{0,2}, s_{0,3}$  ), (  $s_{1,1}, s_{1,2}, s_{1,3}$  ) )

6           approx\_path =

7               state\_sets

8                $P_{0,1}^S \quad P_{1,1}^S \quad P_{2,1}^S$

9                $P_{0,2}^S \quad P_{1,2}^S \quad P_{2,2}^S$

10               $P_{0,3}^S \quad P_{1,3}^S \quad P_{2,3}^S$

11              input\_sets

12               $P_{0,1}^U \quad P_{1,1}^U \quad P_{2,1}^U$

13               $P_{0,2}^U \quad P_{1,2}^U \quad P_{2,2}^U$

14              max\_time = 2

15              spawn a Simulation, with the old parameters, Process ID#15 (not shown in  
16              Figure 8)

17              A one step simulation of  $FSM_{verify}$  is performed to produce a next\_state. A

18              next state  $s_{2,1}, s_{2,2}, s_{2,3}$  is found by using a randomly selected input

19              combination contained in  $P_{0,1}^U, P_{0,2}^U$  in combination with  $s_{1,1}, s_{1,2}, s_{1,3}$ .

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

43/50

## Figure 8L

From the parameters passed to simulation, the following new parameters are determined:

```
new_actual_path = ((s0,1, s0,2, s0,3), (s1,1, s1,2, s1,3), (s2,1, s2,2, s2,3))  
new_approx_path =  
    state_sets  
        P0,1S  
        P0,2S  
        P0,3S  
    input_sets  
        P0,1U  
        P0,2U  
max_time = 0
```

Notes:

The column of state\_sets only contains  $s_{2,1}, s_{2,2}, s_{2,3}$ .

The column of input\_sets accepts any input combination.

spawn a Forward Approximation, with the new parameters, Process ID#16.

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

44/50

## Figure 8M

1 Forward Approximation: Process ID#16

2

3 The following values are unchanged by forward\_approx:

4     actual\_path = ( (s<sub>0,1</sub>, s<sub>0,2</sub>, s<sub>0,3</sub>), (s<sub>1,1</sub>, s<sub>1,2</sub>, s<sub>1,3</sub>), (s<sub>2,1</sub>, s<sub>2,2</sub>, s<sub>2,3</sub>) )

5

6     local priority = 1

7

8 The following values are after having been changed by forward\_approx:

9     approx\_path =

10         state\_sets

11             P<sub>0,1</sub><sup>S</sup>     P<sub>1,1</sub><sup>S</sup>

12             P<sub>0,2</sub><sup>S</sup>     P<sub>1,2</sub><sup>S</sup>

13             P<sub>0,3</sub><sup>S</sup>     P<sub>1,3</sub><sup>S</sup>

14         input\_sets

15             P<sub>0,1</sub><sup>U</sup>     P<sub>1,1</sub><sup>U</sup>

16             P<sub>0,2</sub><sup>U</sup>     P<sub>1,2</sub><sup>U</sup>

17

18         max\_time = 1

19

20 Notes:

21         spawn Bidirectional Approximation, Process ID#17

## Figure 8N

1    Bidirectional Approximation: Process ID#17

2

3    The following values are unchanged by bidirectional\_approx:

4         $\text{actual\_path} = ((s_{0,1}, s_{0,2}, s_{0,3}), (s_{1,1}, s_{1,2}, s_{1,3}), (s_{2,1}, s_{2,2}, s_{2,3}))$

5

6        local priority = 1

7

8    The following values are after having been changed by bidirectional\_approx:

9         $\text{approx\_path} =$

10              state\_sets

11               $P_{0,1}^S \quad P_{1,1}^S$

12               $P_{0,2}^S \quad P_{1,2}^S$

13               $P_{0,3}^S \quad P_{1,3}^S$

14              input\_sets

15               $P_{0,1}^U \quad P_{1,1}^U$

16               $P_{0,2}^U \quad P_{1,2}^U$

17

18        max\_time = 1

19

20    Notes:

21        state\_sets of time 1 are narrowed to their intersection with  $E_1^S, E_2^S, \dots, E_n^S$ .

22

23        state\_sets of time 0 cannot be further narrowed by forward image or  
24        reverse image computations.

25

26        input\_sets of times 0 may be narrowed by reverse image computations.

27

28        spawn Simulation, Process ID#18

METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

46/50

## Figure 8O

1   Simulation: Process ID#18

2  
3   proceeds according to the following major steps:  
4       begins with following passed parameters and values:  
5           local priority = 1

6  
7       actual\_path = ( (  $s_{0,1}, s_{0,2}, s_{0,3}$  ), (  $s_{1,1}, s_{1,2}, s_{1,3}$  ), (  $s_{2,1}, s_{2,2}, s_{2,3}$  ) )  
8

9       approx\_path =  
10          state\_sets

11               $P_{0,1}^S$      $P_{1,1}^S$

12               $P_{0,2}^S$      $P_{1,2}^S$

13               $P_{0,3}^S$      $P_{1,3}^S$

14          input\_sets

15               $P_{0,1}^U$      $P_{1,1}^U$

16               $P_{0,2}^U$      $P_{1,2}^U$

17  
18       max\_time = 1  
19

20       spawn a Simulation, with the old parameters (not shown in Figures 7 or 8)  
21

22       A one step simulation of  $FSM_{verify}$  is performed to produce a next\_state. A  
23       valid next state  $s_{3,1}, s_{3,2}, s_{3,3}$  is found by using a randomly selected input  
24       combination contained in  $P_{0,1}^U, P_{0,2}^U$  in combination with  $s_{2,1}, s_{2,2}, s_{2,3}$ .  
25

26       Since new end of path  $s_{3,1}, s_{3,2}, s_{3,3}$  is in  $E_1^S, E_2^S, \dots, E_n^S$ , end search and return  
27       as value of new\_actual\_path: ( (  $s_{0,1}, s_{0,2}, s_{0,3}$  ), (  $s_{1,1}, s_{1,2}, s_{1,3}$  ), (  $s_{2,1}, s_{2,2}, s_{2,3}$  ),  
28       (  $s_{3,1}, s_{3,2}, s_{3,3}$  ) ).

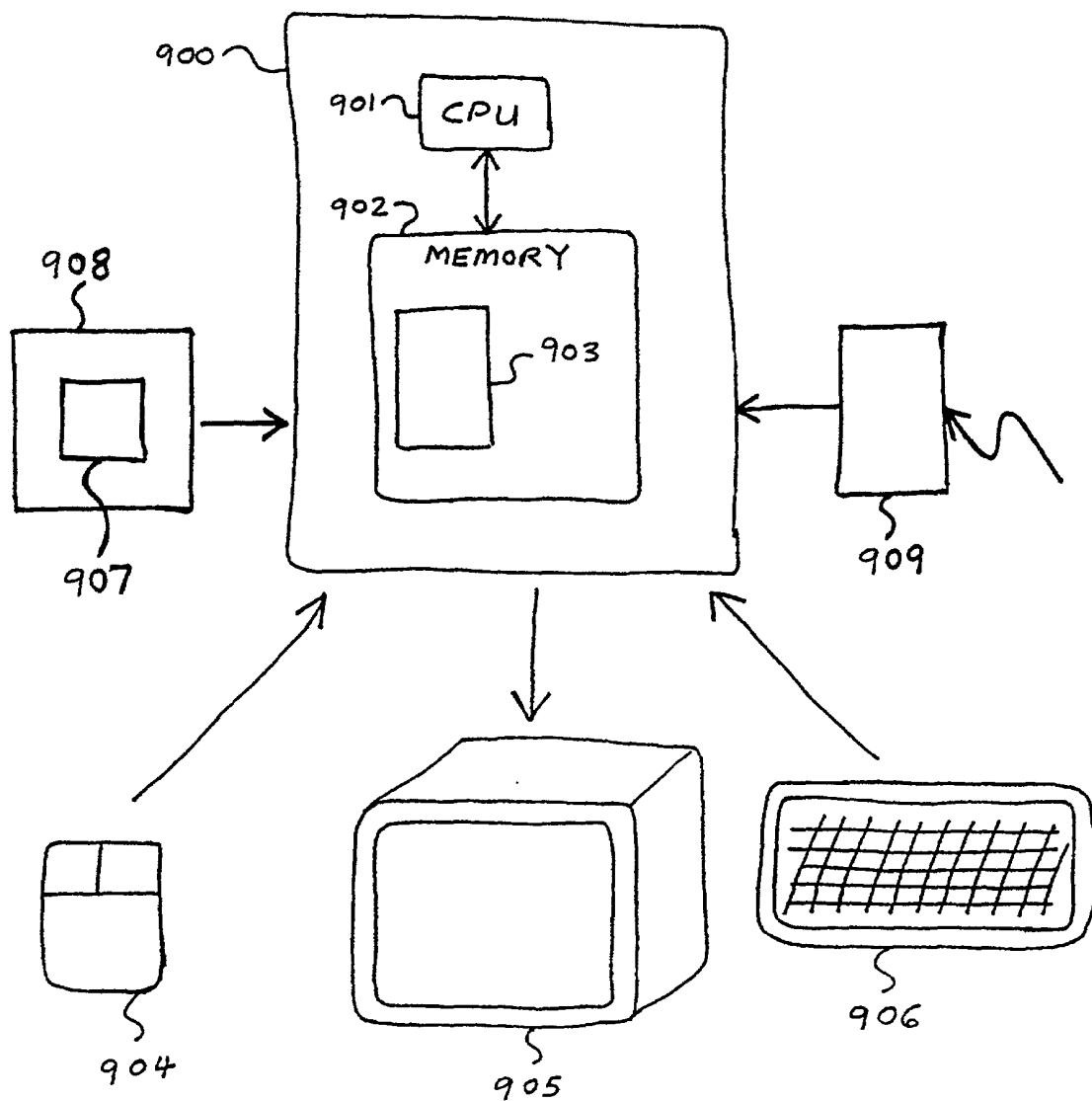
METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

47/50

FIG. 9



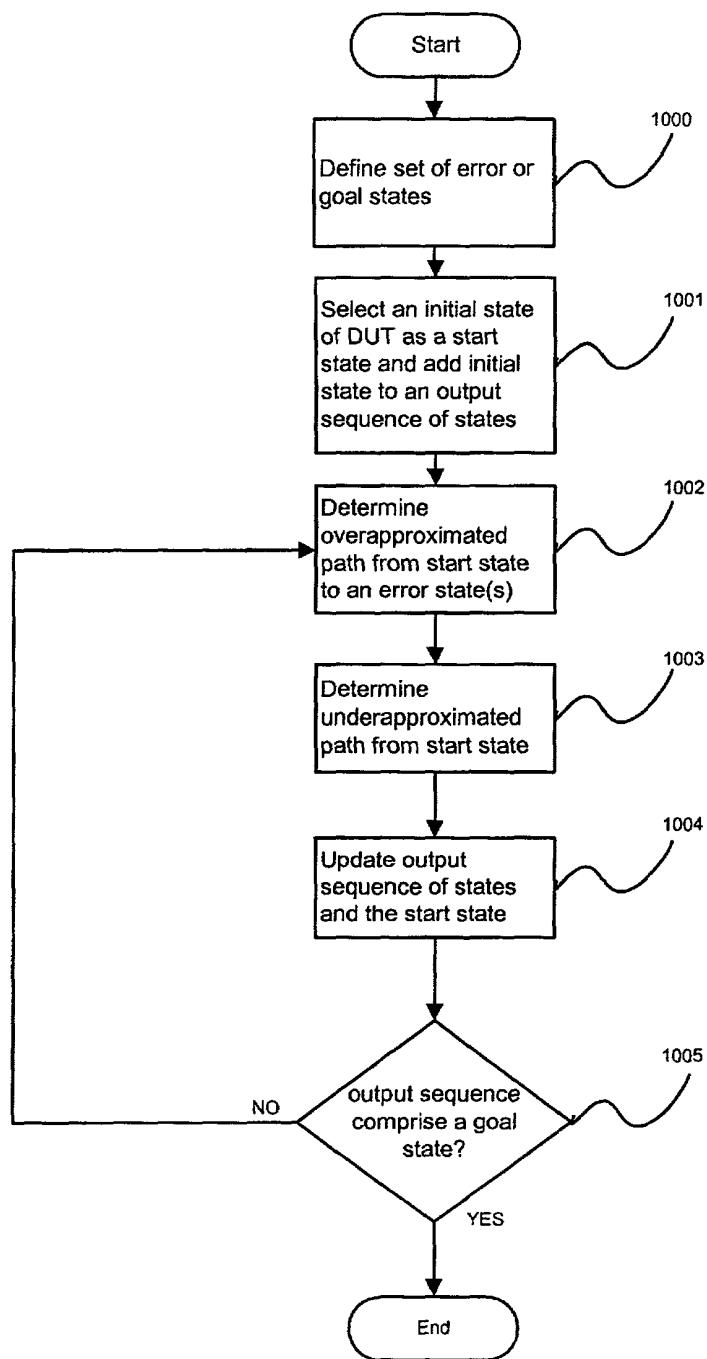
METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

48/50

Figure 10



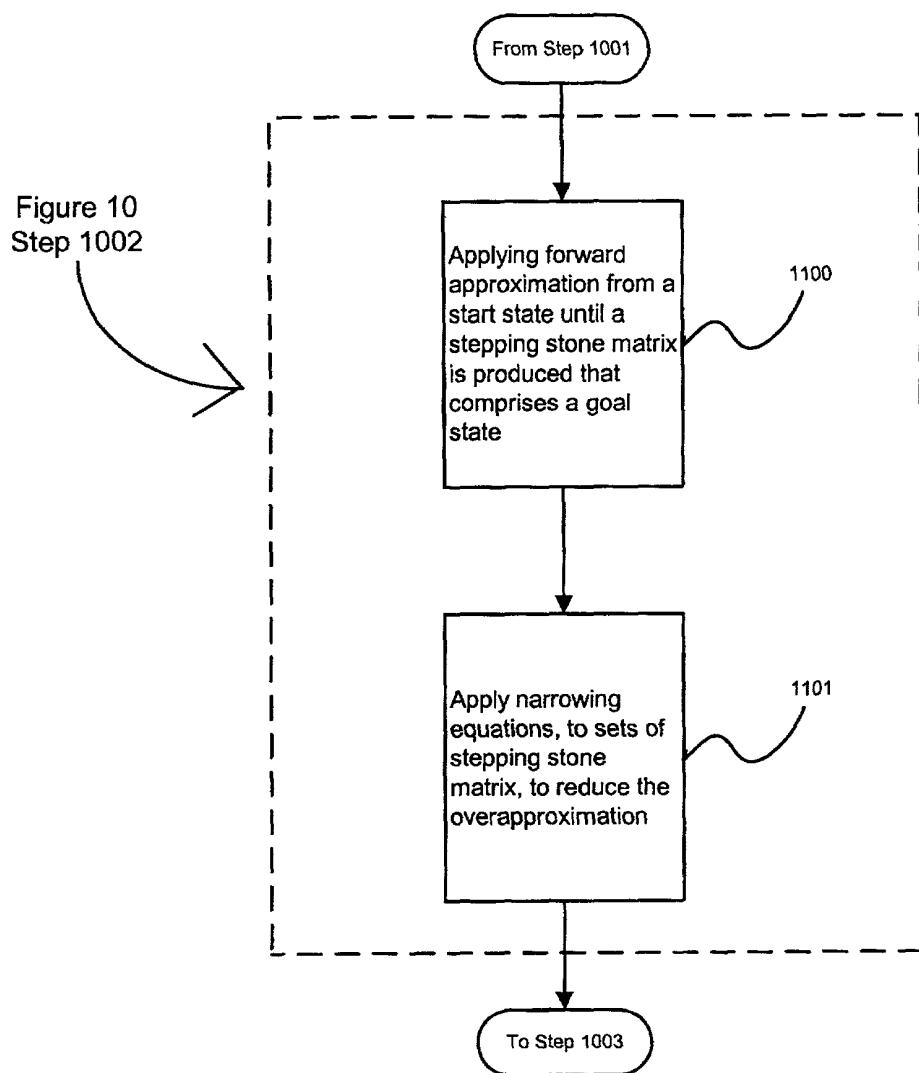
METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

49/50

Figure 11



METHOD AND APPARATUS FOR FORMALLY CONSTRAINING  
RANDOM SIMULATION

J.H. Kukula, et al.

06816.0036

50/50

Figure 12

